A Robust Two-Level Incomplete Factorization for (Navier–)Stokes Saddle Point Matrices

Wubs, Fred W.; Thies, Jonas

Published in:
SIAM Journal on Matrix Analysis and Applications

DOI:
10.1137/100789439

Publication date:
2011

SIAM J. MATRIX ANAL. APPL.
Vol. 32, No. 4, pp. 1475–1499

A ROBUST TWO-LEVEL INCOMPLETE FACTORIZATION FOR (NAVIER–)STOKES SADDLE POINT MATRICES*

FRED W. WUBS† AND JONAS THIES‡

Abstract. We present a new hybrid direct/iterative approach to the solution of a special class of saddle point matrices arising from the discretization of the steady incompressible Navier–Stokes equations on an Arakawa C-grid. The two-level method introduced here has the following properties: (i) it is very robust, even close to the point where the solution becomes unstable; (ii) a single parameter controls fill and convergence, making the method straightforward to use; (iii) the convergence rate is independent of the number of unknowns; (iv) it can be implemented on distributed memory machines in a natural way; (v) the matrix on the second level has the same structure and numerical properties as the original problem, so the method can be applied recursively; (vi) the iteration takes place in the divergence-free space, so the method qualifies as a "constraint preconditioner"; (vii) the approach can also be applied to Poisson problems. This work is also relevant for problems in which similar saddle point matrices occur, for instance, when simulating electrical networks, where one has to satisfy Kirchhoff's conservation law for currents.

Key words. saddle point problem, indefinite matrix, $\mathcal{F}$-matrix, incomplete factorization, grid-independent convergence, Arakawa C-grid, incompressible (Navier–)Stokes equations, constraint preconditioning, electrical networks

AMS subject classifications. 65F08, 65F10, 65F50, 76D07

DOI. 10.1137/100789439

1. Introduction.

Presently, a typical computational fluid dynamics (CFD) problem may involve millions of unknowns. They represent velocities and pressures on a grid and are determined by solving a large sparse linear system of equations. Robust numerical methods are needed to achieve high fidelity. Therefore one often resorts to direct (sparse) solvers. In general such a method does not fail as long as the used precision is enough to handle the posedness of the problem. However, there are two disadvantages to direct methods. First, the amount of memory required for the factorization is not linear in the number of unknowns, and when increasing the problem size one may encounter memory limitations sooner than expected due to fill generated in the factors. Second, all the new elements in the factorization have to be computed, so that the computing time grows sharply, too. This holds especially for three-dimensional (3D) problems, where the computational complexity of direct methods for partial differential equations (PDEs) grows with the square of the number of unknowns.

For this reason one has to resort to iterative methods for very large applications. Such methods perform a finite number of iterations to yield an approximate solution. In theory the accuracy achieved increases with the number of iterations performed. However, iterative methods are often not robust for complex problems. The iteration

*Received by the editors March 19, 2010; accepted for publication (in revised form) by A. J. Wathen September 16, 2011; published electronically December 8, 2011.
http://www.siam.org/journals/simax/32-4/78943.html
†Johan Bernoulli Institute of Mathematics and Computing Science, University of Groningen, P.O. Box 407, 9700 AK Groningen, The Netherlands (f.w.wubs@rug.nl).
‡Centre for Interdisciplinary Mathematics, Department of Mathematics, Uppsala University, P.O. Box 480, 751 06 Uppsala, Sweden (jonas@math.uu.se). The research of this author was mostly supported by the Netherlands Organization for Scientific Research (NWO), through contract ALW854.00.028.

process may stall or diverge and the final approximation may be inaccurate. Furthermore they often require custom numerics such as preconditioning techniques to be efficient.

The hybrid direct/iterative approach presented here seeks to combine the robustness of direct solvers with the memory and computational efficiency of iterative methods. It is based on the direct method recently developed for the Stokes $\mathcal{F}$-matrix in [19], which has the property that the fill does not increase in the "gradient" and "divergence" part of the matrix. To extend this to an incomplete factorization preconditioner one only has to drop velocity-velocity couplings to limit the amount of fill. We perform a nonoverlapping domain decomposition of the grid, and eliminate the interior velocities using a direct method. For the remaining variables a Schur complement problem has to be solved, which we do by a Krylov subspace method preconditioned by a novel incomplete factorization preconditioner.

In this paper we start out by giving a survey of previous research in section 2. In section 3 we will describe the problem in more detail and review the direct method developed in [19]. In section 4 we will introduce the proposed iterative procedure based on this direct method. In section 5 we present numerical results for a series of increasingly complex CFD problems: the Poisson, Darcy, Stokes, and Navier–Stokes equations. We conclude in section 6 by summarizing the method and results and giving an outlook on future work.

2. Survey of previous work. In [2] a survey is given of methods currently in use to solve linear systems from fluid flow problems. In many cases saddle point problems can be solved efficiently by a Krylov subspace iteration [25] combined with appropriate preconditioning [3, 2, 18, 8, 15, 9]. Often a segregated approach is used, i.e., the velocities are solved independently from the pressures. This results in inner and outer iterations, the former for the independent systems for velocities and pressures, and the latter to iterate to the solution of the whole system. We advocate a fully coupled approach.

The idea of combining direct and iterative methods has been used in [13] and [10] to solve general sparse linear systems arising from the discretization of scalar PDEs. As in this paper, they reduce the problem to a Schur complement system on the separators of a domain decomposition. The Schur complement system is solved iteratively using an ILU factorization. As the structural and numerical properties are not explicitly preserved, robustness and grid-independence cannot be ascertained for indefinite problems.

Recently, de Niet and Wubs [19] proposed a direct method for the solution of $\mathcal{F}$-matrices, of which the incompressible Stokes equations on an Arakawa C-grid are a special case. This special purpose method reduces fill and computation time while preserving the structure of the equations during the elimination. It still suffers from the weaknesses of direct methods, but only the number of velocity-velocity couplings increases, not the number of velocity-pressure couplings. We believe that a better understanding of the $\mathcal{F}$-matrices will lead to generalizations that are of interest to a broader class of indefinite problems and note that there are applications outside the field of fluid mechanics, e.g., in electronic circuit simulations [24], which lead to $\mathcal{F}$-matrices. We refer to section 6 for a discussion of generalizations.

For incompressible flow one has to satisfy an incompressibility constraint: the velocity should be divergence-free. We remark that our iterative technique does not violate the divergence constraint and therefore belongs to the class of "constraint preconditioners" [16]. For details, see section 4.5.

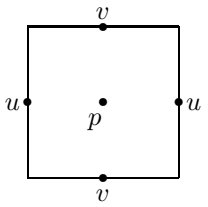

FIG. 1. *Positioning of velocity (u, v) and pressure (p) variables in the C-grid.*

3. $\mathcal{F}$-matrices and the direct solution method. In this paper we study the solution of the equation

$$Kx = b, \tag{1}$$

where $K \in R^{(n+m)\times(n+m)}$ $(n \geq m)$ is a saddle point matrix that has the form

$$K = \begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix}, \tag{2}$$

with $A \in R^{n\times n}$, $B \in R^{n\times m}$. Special attention is given to a class of saddle point matrices known as $\mathcal{F}$-matrices. We start out by defining the gradient matrix in which the $\mathcal{F}$-matrix is expressed.

DEFINITION 3.1. *A gradient-matrix has at most two nonzero entries per row and its row sum is zero.*

We have chosen the name gradient-matrix, because this type of matrix typically results from the discretization of a pressure gradient in flow equations. It is important to note that the definition allows a gradient-matrix to be nonsquare. Now we can define the $\mathcal{F}$-matrix.

DEFINITION 3.2. *A saddle point matrix* (2) *is called an $\mathcal{F}$-matrix if A is positive definite and B is a gradient-matrix.*

The definition is due to Tůma [23]. $\mathcal{F}$-matrices occur in various fluid flow problems where Arakawa A-grids (collocated) or C-grids (staggered; see Figure 1) are used. For example, in [1] the discretization of Darcy's equation in groundwater flow results in an $\mathcal{F}$-matrix. They also occur in electronic network simulations [24].

3.1. The algorithm for the direct approach. In this section we explain the direct method which forms the basis for the iterative method. Many of the standard algorithms have in common that they compute a fill-reducing ordering for K and then somehow adapt it to make it feasible: a factorization is feasible if it does not break down due to a zero pivot. The delay of elimination (through pivoting) will give an increase in computing time and may lead to increased fill in the factors. To preclude this inefficiency, the following algorithm was proposed in [19]. Suppose the sets of all velocities and pressures are denoted by V and P, respectively. The respective elements will be called V-nodes and P-nodes. $F(A)$ denotes the fill pattern of the matrix A.

ALGORITHM 1. *To compute a feasible fill-reducing ordering for the saddle point matrix K:*

1. *Compute a fill-reducing ordering for the V-nodes based on $F(A) \cup F(BB^T)$.*

2. *Insert the P-nodes into the ordering according to Rule* 1.

RULE 1. *During Gaussian elimination with K, whenever a V-node is to be eliminated which is connected to a P-node, these nodes are eliminated together using a 2×2 pivot.*

With this rule we get as many 2×2 pivots as there are P-nodes. Only if due to elimination a V-node becomes totally disconnected from P it can be eliminated on its own.

As all P-nodes are eliminated together with a V-node in pivots of the form

$$\begin{pmatrix} \alpha & \beta \\ \beta & 0 \end{pmatrix},$$

the factorization is always feasible and additional pivoting is not required. The reduced matrix obtained after elimination is called the Schur complement below.

The above method has structure preserving properties which we list in the theorems below. The first two are taken from [19], where they were proved for symmetric positive definite A. Along the same lines they can be proved for nonsymmetric positive definite A.

THEOREM 3.3. *If K is an $\mathcal{F}$-matrix, all Schur complements $K^{(l)}$ are $\mathcal{F}$-matrices.*

This means that the A part will remain positive definite and the B part will have at most two entries per row in any step of the elimination. The latter allows us to keep the B part exact during the incomplete factorization.

THEOREM 3.4. *The B part in all Schur complements is independent of the size of the entries in the A part.*

THEOREM 3.5. *If initially B has entries with magnitude one, then this will remain so during the elimination.*

THEOREM 3.6. *If a P-node is not eliminated together with the first V-node it is attached to, the next Schur complement will not be an $\mathcal{F}$-matrix.*

Proof. Consider the matrix in (3) in the next section. It is clear that using only α as pivot will give a contribution in the zero block. ∎

4. Structure preserving incomplete factorization. In this section we want to develop an incomplete factorization based on the direct method described so far. First we will introduce the domain decomposition we use and then we will illustrate that simply applying a dropping strategy to the A part may not give the desired result when there are couplings to P-nodes. We then proceed to develop a combination of orthogonal transformations and dropping that leads to grid-independent convergence, limits fill-in, and keeps the divergence constraint intact.

Assumption. For this section we will assume that the entries in B have equal magnitude. This is not a restriction because it can be achieved by scaling the rows of an arbitrary gradient matrix B. If DB gives the desired matrix, our new matrix will be

$$\begin{bmatrix} DAD & DB \\ B^T D & O \end{bmatrix}.$$

Observe that the postscaling means that the V-nodes will be scaled. For Navier–Stokes on a stretched grid (see section 5.4) the scaling is such that we get as new unknowns the fluxes through the control cell boundaries.

4.1. Domain decomposition. The first step of the proposed method is to assign the unknowns $u_i \in V \cup P$ to either of two sets I (interior variables) and S

(separator variables). This ordering step yields the desired Schur complement for the separator variables when eliminating the interior variables. The corresponding algorithm is given in Algorithm 2.

ALGORITHM 2. *To compute an elimination ordering, we have the following.*

1. *Domain decomposition of the V-nodes based on $F(A) \cup F(BB^T)$, giving n_{SD} subdomains Ω_k.*
2. *Assign each P-node p_j to the subdomain containing the majority of V-nodes $v_i : B_{ij} \neq 0$.*
3. *Let $S_{V,k} = \{v_i \in V \cap \Omega_k : \exists v_j \in \Omega_{l>k} : K_{ij} \neq 0\}$ for $k = 1, \ldots, n_{SD}$ consecutively. $S_V := \bigcup_{k=1}^{n_{SD}} S_{V,k}$.*
4. *Let $I_V = V \backslash S_V$.*
5. *Let S_P contain an arbitrary P-node $p_j \in \Omega_k, k = 1, \ldots, n_{SD}$ such that $B_{ij} \neq 0$ for some $v_i \in I_V \cap \Omega_k$, as well as any P-nodes $p_j : B_{ij} = 0 \ \forall v_i \in I_V$.*
6. *Let $I_P = P \backslash S_P$.*
7. *Let $I = I_V \cup I_P$ and $S = S_V \cup S_P$.*

Step 1 can be seen as a nested dissection ordering (see [11]) as may be used in step 1 of Algorithm 1, stopped at a certain subdomain size (see also [22] in the paragraph "Schur complement systems" starting on page 262). This can be done by applying a graph-partitioning method like Metis [14] or similar libraries to $F(A) \cup F(BB^T)$. For this study we use a manual partitioning into equally sized square subdomains. (For the Navier–Stokes equations we used a stretched grid, so in that case they are not square and equally sized in physical space but in the number of unknowns). For step 2, note that every P-node is connected to four V-nodes.

We then introduce a minimal overlap in step 3. Introducing an ordering on the subdomains here makes sure that there is only one layer of separator V-nodes between subdomains (in the case of periodic boundary conditions this ordering has to be slightly adjusted). The S_V nodes are complemented by an arbitrary single P-node per subdomain and any P-nodes which connect only to separator velocities. This makes sure that the matrix associated with interior variables is nonsingular (in physical terms the pressure level inside the subdomains is fixed and any grid cell with only separator velocities retains its P-node). We remark that

(i) we used horizontal and vertical separators as depicted for two domains in Figure 2. A better choice may be to use skew separators ($\pm 45°$), leading to about half the V nodes on the separator for subdomains of similar size. Both approaches yield the same number of V nodes with couplings to P nodes in the Schur complement, and we chose for ease of programming here;

(ii) we use the decomposition primarily for numerical reasons and the number of subdomains will typically be much larger than the number of processors in a parallel computation.

We can now eliminate the interior variables, leading to a Schur complement problem for the separator velocities and remaining pressures. The remainder of this section is devoted to constructing an incomplete factorization preconditioner for this Schur complement, so that it can be solved efficiently by a Krylov subspace method.

4.2. The dropping problem. Consider the following matrix, which occurs in any elimination step with a 2×2 pivot:

$$\left[\begin{array}{cc|cc} \alpha & \beta & a^T & b^T \\ \beta & 0 & \hat{b}^T & 0 \\ \hline a & \hat{b} & \hat{A} & \hat{B} \\ b & 0 & \hat{B}^T & O \end{array}\right]. \tag{3}$$

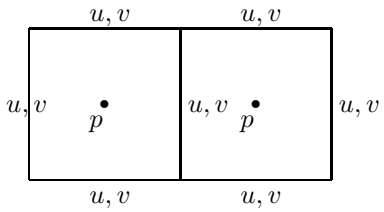

FIG. 2. *Velocity separators (u, v) and pressure per domain (p) in a 2-domains case.*

When performing the elimination step, a multiple of $\hat{b}\hat{b}^T$ is added to $\hat{A}$. This does not introduce new fill if $\hat{A}$ is dense. But if we replaced $\hat{A}$ by a sparse matrix by dropping, the matrix would be filled again as $\hat{b}$ is typically dense.

This is a common phenomenon. Consider, for example, the two-domain case in Figure 2. After eliminating the interior variables, many of the V-nodes on the separator are coupled to the two remaining P-nodes. Assume that we drop all connections between the V-nodes on the separator, so in the above matrix (3), $\hat{A}$ is replaced by its diagonal, and a becomes zero; $\hat{b}$ is a dense vector, $\hat{B}$ has an associated dense column with opposite sign, and b^T has a nonzero at the same column position with sign opposite to that of β. When eliminating one "V-node P-node" pair, all the V-nodes on the separator become detached from P and $\hat{A}$ becomes dense.

From the above we learn that we should try to get more zeros into $\hat{b}$. Or stated otherwise, we should try to decouple the V-nodes on the separator from the P-nodes as far as possible.

4.3. Orthogonal operators to decouple V- and P-nodes. One idea to get rid of unwanted pressure couplings is to simply drop them. However, the fill in the B-part is already modest and an exact B-part is attractive, as discussed in section 4.5. Fortunately we can do better. Consider the square domain decomposition (Figure 2), extended periodically so that every subdomain is bounded by four separators from the neighboring subdomains. The Schur complement for the separator velocities and remaining pressures has about the following form (the V- and P-nodes in the corners are neglected here, in practice they form "complete conservation cells" and get a block of their own):

$$\begin{bmatrix} A_{11} & B_1 & A_{12} & A_{13} & O & O \\ B_1^T & O & B_{21}^T & B_{31}^T & O & O \\ A_{21} & B_{21} & A_{22} & O & A_{24} & B_{22} \\ A_{31} & B_{31} & O & A_{33} & A_{34} & B_{32} \\ O & O & A_{42} & A_{43} & A_{44} & B_{42} \\ O & O & B_{22}^T & B_{32}^T & B_{42}^T & O \end{bmatrix} \begin{bmatrix} v_1 \\ p_1 \\ v_2 \\ v_3 \\ v_4 \\ p_2 \end{bmatrix} = \begin{bmatrix} b_{v_1} \\ b_{p_1} \\ b_{v_2} \\ b_{v_3} \\ b_{v_4} \\ b_{p_2} \end{bmatrix}.$$

Here the variables are grouped as follows:

- v_1: V-nodes on a certain separator;
- p_1: the two P-nodes from the adjacent subdomains;
- v_2 and v_3: V-nodes from other separators around these subdomains;
- v_4 and p_2: remaining V- and P-nodes in the Schur complement.

As B_1 contains only two dense columns, equal up to a sign, we can define an orthogonal transformation H (e.g., a Householder reflection) such that $H^T B_1$ has only entries on a single row. Applying H to the first block row and column from left and right, we

obtain the following system:

$$\begin{bmatrix} H^TA_{11}H & H^TB_1 & H^TA_{12} & H^TA_{13} & O & O \\ (H^TB_1)^T & O & B_{21}^T & B_{31}^T & O & O \\ A_{21}H & B_{21} & A_{22} & O & A_{24} & B_{22} \\ A_{31}H & B_{31} & O & A_{33} & A_{34} & B_{32} \\ O & O & A_{42} & A_{43} & A_{44} & B_{42} \\ O & O & B_{22}^T & B_{32}^T & B_{42}^T & O \end{bmatrix} \begin{bmatrix} H^Tv_1 \\ p_1 \\ v_2 \\ v_3 \\ v_4 \\ p_2 \end{bmatrix} = \begin{bmatrix} H^Tb_{v_1} \\ b_{p_1} \\ b_{v_2} \\ b_{v_3} \\ b_{v_4} \\ b_{p_2} \end{bmatrix}.$$

Applying H is cheap if its defining form is exploited, but generally destroys sparsity. However, the matrices A_{11}, A_{12}, and A_{13} are typically already dense (see Remark 1 below), so not much is lost and we have gained a lot: while the properties of the matrix are preserved exactly, we decoupled all but one of the V-nodes on the separator from the P-nodes. The decoupled ones can be eliminated on their own now.

Remark 1. The fill of A_{11}, A_{12}, and A_{13} depends on the problem at hand. For the two-dimensional (2D) Stokes equations in the absence of the pressure terms we get two decoupled Poisson equations for u and v. In that case nested dissection gives connections between all the variables surrounding a domain. So the matrices A_{11}, A_{12}, and A_{13} are half full (no couplings between u and v). As most pressures are eliminated with the interior velocities, the matrices become dense.

Remark 2. In practice, u and v nodes on a separator may connect to the P-nodes with reversed signs. To ensure robustness we apply separate transforms to each velocity component.

Remark 3. Choosing a Householder transformation may seem arbitrary and not related to the physics of the problem. We may indeed choose other orthogonal transformations with the same effect (some alternatives are proposed at the end of section 4.4). The key is that one of the columns of H—up to a normalizing factor—should be the vector e with all entries equal to one. This yields the sum of all the fluxes through the interface, so there will be a new variable that represents the entire flux through the interface. The other new variables represent fluxes through the interface that are on average zero.

Remark 4. Instead of scaling the vectors in H to unit length, we scale them to the length $m = ||e||_2$ of the vector e defining H. In that case the inverse of H is $\frac{1}{m}H^T$.

Remark 5. Although not necessary for the decoupling process, we also apply an orthogonal transformation to V-nodes that are not coupled to a P-node in the first place. This is important for the dropping strategy proposed in the next section.

The situation depicted in (3) now occurs only once per separator and velocity component, namely for the V-node still coupled to the P-nodes. Because of the transformation $\hat{b}$ is now zero, and no fill is generated.

So far we have not made any approximations, and while we have zeroed out most of the V-node/P-node couplings, a dropping strategy has to be applied in the V-V part to get a sparse preconditioner for the Schur complement. However, the Householder transformation combined with standard dropping techniques for the symmetric positive definite (SPD) case will generally not lead to grid independent convergence. This requires that the approximation is spectrally equivalent to the original matrix. We will consider a new way of dropping in the next section which has this property.

4.4. Dropping strategy and convergence analysis. The general idea of the approximation is the following. We replace the flux through grid cell faces forming a separator by the combined flux through that separator (see Remark 3 in the previous

section). Then we try to reduce the problem of finding all separator velocities by dropping and elimination to the related problem of finding the new fluxes (or summed velocities). This reduced problem can still be understood in terms of conservation of mass and momentum and its form is very similar to the original problem.

Let us consider an orthogonal operator that is more intuitive than the Householder transformation. Suppose e is a vector with all ones and C is an orthogonal extension of e such that the length of every column is the same. Define a square matrix

$$H = [C, e],$$

which is orthogonal up to a constant factor (see Remark 4 in the previous section). This operator is applied to the velocity component in normal direction on the separator. These velocities have the same sign for the connection to the pressure and therefore again only one row remains in $H^T B_1$. The first component of $H^T v$ will be the sum of the components of v; we will call this a V_Σ-node from now on. The following lemma and its corollary play a key role in devising a dropping strategy.

LEMMA 4.1. *Principal submatrices of an (S)PD-matrix are (S)PD.*

COROLLARY 4.2.

$$\textit{If } \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \textit{ is (S)PD, then } \begin{bmatrix} A_{11} & O \\ O & A_{22} \end{bmatrix} \textit{ is (S)PD.}$$

Since we only make approximations in the A part of the matrix K, we have the following lemma.

LEMMA 4.3. *If A is SPD, the condition number of the preconditioned K matrix is bounded by the condition number of the preconditioned A, where as preconditioner an SPD approximation of A is used.*

Proof. Consider the generalized eigenvalue problem

$$\begin{bmatrix} A - \lambda\tilde{A} & (1-\lambda)B \\ (1-\lambda)B^T & O \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0, \tag{4}$$

where $\tilde{A}$ denotes an SPD approximation of A. We see that for $\lambda \neq 1$ ($\lambda = 1$ is clearly an eigenvalue) we can scale the border by any constant. So the eigenvalue problem is in fact an eigenvalue problem restricted to the kernel of the divergence (or constraint) operator B^T. Suppose Q is an orthogonal basis for the kernel of B^T, then we have to find the eigenvalues of the pencil $(Q^T A Q, Q^T \tilde{A} Q)$. Now

$$\begin{aligned} \lambda_{\min}(A, \tilde{A}) &= \min_x \frac{(x, Ax)}{(x, \tilde{A}x)} \leq \min_y \frac{(y, Q^T A Q y)}{(y, Q^T \tilde{A} Q y)} \leq \frac{(y, Q^T A Q y)}{(y, Q^T \tilde{A} Q y)} \\ &\leq \max_y \frac{(y, Q^T A Q y)}{(y, Q^T \tilde{A} Q y)} \leq \max_x \frac{(x, Ax)}{(x, \tilde{A}x)} = \lambda_{\max}(A, \tilde{A}). \end{aligned}$$

Hence, the eigenvalues of the preconditioned K are bounded by the eigenvalues of the preconditioned A, which leads to the result. ∎

These lemmas set the ground for further reasoning that will lead to grid-independent convergence. In the remainder of this section we assume that A is symmetric and positive definite. Let us extend H with an identity for the unknowns that are not transformed and write $H = [H_1, H_2]$, where

$$H_1 = \begin{bmatrix} C \\ 0 \end{bmatrix}, \quad H_2 = \begin{bmatrix} e & 0 \\ 0 & I \end{bmatrix}.$$

The transformed matrix is given by

$$H^T AH = \begin{bmatrix} H_1^T AH_1 & H_1^T AH_2 \\ H_2^T AH_1 & H_2^T AH_2 \end{bmatrix}. \tag{5}$$

Here $H_2^T AH_2$ is a Galerkin approximation of A and hence it can be viewed as a discretization on a coarser grid (in fact it is an aggregation similar to that used by Notay in [20], albeit Notay applies the aggregation directly to the discretized PDE whereas we apply it to its Schur complement on the separators). If A is obtained from a stable discretization of a second-order differential operator, then $H_1^T AH_1$ has a condition number independent of the mesh size if the dimension of C is fixed (i.e., if the length of the separator is fixed).

We assume that our matrix satisfies the inf-sup condition

$$\max_v \frac{(p, B^T v)^2}{(v, Av)} > c(p, p) \quad \text{for all } p \perp e, \tag{6}$$

and for the Stokes problem also boundedness on BB^T in terms of A:

$$(B^T x, B^T x) < c_2(x, Ax) \quad \text{for all } x. \tag{7}$$

It can be shown that these conditions are satisfied for the discretization of the Stokes equation we consider in this paper.

The proof follows now in two main steps: (i) Show that the condition number of the preconditioned A part of the Schur complement of the Stokes problem is bounded by the preconditioned Schur complement of the Laplace problem and next (ii) show that the preconditioned Schur complement of the Laplace problem has a condition number independent of the mesh.

For the first part we need the following lemma, which is just a reformulation of Lemma 9.10 in [22] in matrix form.

LEMMA 4.4. *Let K satisfy the conditions* (6) *and* (7). *And let A_S denote the A part of the Stokes Schur complement acting on the separators and A_L the Laplace Schur complement on the same separators. Then it holds that*

$$\alpha(x, A_S x) \le (x, A_L x) \le (x, A_S x),$$

where α is a positive constant.

Proof. The proof can be given precisely along the line indicated in [6], using the conditions in the lemma. One minor point is to show that the Schur complement matrix we have here and the Schur complement matrix one gets after a domain decomposition are identical. The point in the latter case is that in each subdomain the pressure is split into a constant pressure plus a perturbation perpendicular to that. After eliminating the internal nodes, only the coupling to the constant parts remains. These Schur complements are indeed the same. ∎

This relation helps to bound the condition number of the A part of the preconditioned Schur complement from the Stokes problem by that of the preconditioned Schur complement from the Laplace problem.

LEMMA 4.5. *Let $\tilde{A}_S$ and $\tilde{A}_L$ be the preconditioners of A_S and A_L, respectively. Then $\kappa_*(\tilde{A}_S^{-1} A_S) \le \kappa_*(\tilde{A}_L^{-1} A_L)/\alpha^2$, where κ_* is the spectral condition number.*

Proof. From Lemma 4.4 we know that

$$\alpha(H_i x, A_S H_i x) \le (H_i x, A_L H_i x) \le (H_i x, A_S H_i x) \text{ for } i = 1, 2.$$

Moreover, $\alpha(Hx, A_S Hx) \leq (Hx, A_L Hx) \leq (Hx, A_S Hx)$. Now the preconditioner is (implicitly) defined by

$$H^T \tilde{A}_S H = \mathrm{diag}[(H^T A_S H)_{11},\ (H^T A_S H)_{22}] = \mathrm{diag}[(H_1^T A_S H_1),\ (H_2^T A_S H_2)],$$

and likewise for A_L. Hence, we have $\alpha(Hx, \tilde{A}_S Hx) \leq (Hx, \tilde{A}_L Hx) \leq (Hx, \tilde{A}_S Hx)$, and consequently also $\alpha(x, \tilde{A}_S x) \leq (x, \tilde{A}_L x) \leq (x, \tilde{A}_S x)$. Using this we find the following relation:

$$\alpha \frac{(x, A_S x)}{(x, \tilde{A}_S x)} \leq \frac{(x, A_L x)}{(x, \tilde{A}_L x)} \leq \frac{1}{\alpha} \frac{(x, A_S x)}{(x, \tilde{A}_S x)}.$$

Thus we can express the eigenvalues of the preconditioned Schur complement matrix from the Stokes problem in those of the preconditioned Schur complement matrix from the Laplace problem. ∎

The problem is now reduced to finding the condition number of the preconditioned Laplace Schur complement matrix. From now on we assume that A is the Laplace Schur complement and define $\check{A} = H^T A H$, where H is the transformation introduced earlier but now extended to the whole grid. Then we can split it into $\check{A} = \check{D} + \check{R}$, where $\check{D}$ is the preconditioner and $\check{R}$ the remaining part. Thus, we have to bound the condition number of $\check{D}^{-1}A = \check{D}^{-1}(\check{D} + \check{R}) = I + \check{D}^{-1}\check{R}$. The eigenvalues of $\check{D}^{-1}\check{R}$ should be larger than -1, which is always the case for A SPD, since then $(v, \check{D}v) + (v, \check{R}v) > 0$, and hence dividing by $(v, \check{D}v)$ gives $(v, \check{R}v)/(v, \check{D}v) > -1$. To get grid-independent convergence, the eigenvalues of $\check{D}^{-1}\check{R}$ must be in an interval to the right from -1 independent of the mesh size.

Without loss of generality, we consider the case with two separators to show what is happening in the matrix. Here A_{11} and A_{22} are the matrices associated with these two separators. Now we pre- and postmultiply by H^T and H, respectively, with $H = \mathrm{diag}([C_1 e_1], [C_2, e_2], I, I])$. Next we reorder the matrix such that we get

$$\check{R} = \begin{bmatrix} 0 & C_1^T A_{12} C_2 & C_1^T A_{11} e_1 & C_1^T A_{12} e_2 & C_1^T A_{13} & 0 \\ C_2^T A_{21} C_1 & 0 & C_2^T A_{21} e_1 & C_2^T A_{22} e_2 & C_2^T A_{23} & C_2^T A_{24} \\ e_1^T A_{11} C_1 & e_1^T A_{12} C_2 & 0 & 0 & 0 & 0 \\ e_2^T A_{21} C_1 & e_2^T A_{22} C_2 & 0 & 0 & 0 & 0 \\ A_{31} C_1^T & A_{32} C_2^T & 0 & 0 & 0 & 0 \\ 0 & A_{42} C_2 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$\check{D} = \begin{bmatrix} C_1^T A_{11} C_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & C_2^T A_{22} C_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & e_1^T A_{11} e_1 & e_1^T A_{12} e_2 & e_1^T A_{13} & 0 \\ 0 & 0 & e_2^T A_{21} e_1 & e_2^T A_{22} e_2 & e_2^T A_{23} & e_2^T A_{24} \\ 0 & 0 & A_{31} e_1 & A_{32} e_2 & A_{33} & A_{34} \\ 0 & 0 & 0 & A_{42} e_2 & A_{43} & A_{44} \end{bmatrix}.$$

Note that this matrix does not have Property A [28]. Therefore we also introduce a

splitting of $\check{A}$ which has this property:

$$\hat{R} = \begin{bmatrix} 0 & 0 & C_1^T A_{11} e_1 & C_1^T A_{12} e_2 & C_1^T A_{13} & 0 \\ 0 & 0 & C_2^T A_{21} e_1 & C_2^T A_{22} e_2 & C_2^T A_{23} & C_2^T A_{24} \\ e_1^T A_{11} C_1 & e_1^T A_{12} C_2 & 0 & 0 & 0 & 0 \\ e_2^T A_{21} C_1 & e_2^T A_{22} C_2 & 0 & 0 & 0 & 0 \\ A_{31} C_1^T & A_{32} C_2^T & 0 & 0 & 0 & 0 \\ 0 & A_{42} C_2 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$\hat{D} = \begin{bmatrix} C_1^T A_{11} C_1 & C_1^T A_{12} C_2 & 0 & 0 & 0 & \\ C_2^T A_{21} C_1 & C_2^T A_{22} C_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & e_1^T A_{11} e_1 & e_1^T A_{12} e_2 & e_1^T A_{13} & 0 \\ 0 & 0 & e_2^T A_{21} e_1 & e_2^T A_{22} e_2 & e_2^T A_{23} & e_2^T A_{24} \\ 0 & 0 & A_{31} e_1 & A_{32} e_2 & A_{33} & A_{34} \\ 0 & 0 & 0 & A_{42} e_2 & A_{43} & A_{44} \end{bmatrix}.$$

Now we define $\check{D}_{aa}$ as the upper 2×2 block of $\check{D}$ and similarly $\check{R}_{aa}$.

LEMMA 4.6. *Let the spectral radius $\rho(\hat{D}^{-1}\hat{R}) \le \eta$ with $\eta < 1$ and let $\sigma(\check{D}_{aa}^{-1}(\check{D}_{aa} + \check{R}_{aa}))$ be contained in $[\alpha, \beta]$ with $\alpha \le 1$ and $\beta \ge 1$. Then the spectrum $\sigma(\check{D}^{-1}\check{A})$ is contained in $[\alpha(1-\eta), \beta(1+\eta)]$ and the condition number of the preconditioned matrix is bounded by $(\beta(1+\eta))/(\alpha(1-\eta))$.*

Proof. The first condition gives $\sigma(\hat{D}^{-1}A) \subset [1-\eta, 1+\eta]$, and the second $\alpha(x, \check{D}_{aa}x) \le (x, (\check{D}_{aa} + \check{R}_{aa})x) = (x, \hat{D}_{aa}x) \le \beta(x, \check{D}_{aa}x)$. For the complete matrices (so x has the size of the order of $\check{A}$ here) we get

$$\alpha(1-\eta)(x, \check{D}x) \le (1-\eta)(x, \hat{D}x) \le (x, \check{A}x) \le (1+\eta)(x, \hat{D}x) \le \beta(1+\eta)(x, \check{D}x),$$

which proves the assertion. □

We now introduce two splittings of A that are important for bounds we need in the proof. Consider first the standard five point stencil of the Laplace equation on an equidistant grid. We reorder the matrix such that all separator nodes occur at the end of the unknowns vector. The system matrix has the form

$$\begin{bmatrix} A_{II} & A_{IS} \\ A_{SI} & A_{SS} \end{bmatrix}, \tag{8}$$

where I stands for interior and S for separator. Now we consider a number of consecutive unknowns on a part of one of the separators. The subblock in A_{SS} associated with these unknowns has the form

$$2I + \begin{bmatrix} 2 & -1 & & & & \\ -1 & 2 & -1 & & & \\ & & \ddots & \ddots & & \\ & & & -1 & 2 & -1 \\ & & & & -1 & 2 \end{bmatrix}.$$

We split this matrix into $2I + \mathrm{diag}(1, 0, \ldots, 0, 1) + E_{ii}$, where

$$E_{ii} = \begin{bmatrix} 1 & -1 & & & & \\ -1 & 2 & -1 & & & \\ & & \ddots & \ddots & & \\ & & & -1 & 2 & -1 \\ & & & & -1 & 1 \end{bmatrix}.$$

We apply this splitting for all separators. Now if we would skip all the E_{ii} from the matrix (8), then the remaining matrix is still positive definite. We know that the Schur complement of a positive definite matrix is again positive definite. So after eliminating the internal nodes, which means subtracting the nonnegative matrix $A_{SI}A_{II}^{-1}A_{IS}$ from the modified A_{SS}, we again end up with a positive definite matrix which we call matrix $\breve{A}$. So in the example of the two separators we have the splitting $A = \breve{A} + \mathrm{diag}(E_{11}, E_{22}, 0, 0)$. Alternatively, if we remove A_{SS} completely, then after eliminating the internal nodes, we end up with a matrix $\dot{A} \equiv -A_{SI}A_{II}^{-1}A_{IS}$, which is nonpositive. Since we need to subscript A_{SS} later on we define $F \equiv A_{SS}$. So the second splitting is $A = \dot{A} + F$. For the standard five point stencil and the orthogonal separators we chose, there are no direct connections in F from unknowns (to be modified) on one separator to those of another. Hence both F_{12} and F_{21} in the example are zero. We remark that such splittings are possible for much more general discretizations.

The following lemma showing that $\mathrm{diag}(E_{11}, E_{22})$ and $\mathrm{diag}(F_{11}, F_{22})$ are spectrally equivalent on a space perpendicular to the constant is very important in the further analysis.

LEMMA 4.7. *For the equidistant discretization of the Laplace equation we have*

$$\begin{aligned}(9)\qquad \alpha(x, \mathrm{diag}(C_1^T F_{11} C_1, C_2^T F_{22} C_2)x) &\le (x, \mathrm{diag}(C_1^T E_{11} C_1, C_2^T E_{22} C_2)x)\\ &\le 2(x, \mathrm{diag}(C_1^T F_{11} C_1, C_2^T F_{22} C_2)x),\end{aligned}$$

where $\alpha = \min_i \alpha_i$, $\alpha_i = \frac{2}{3m_i^2}$, *and* m_i *is the number of unknowns on the separator.*

Proof. Due to the block diagonal form of the occurring matrices we can restrict the discussion to one separator. There

$$F_{ii} = 2I + \begin{bmatrix} 2 & -1 & & & & \\ -1 & 2 & -1 & & & \\ & \ddots & \ddots & \ddots & & \\ & & -1 & 2 & -1 & \\ & & & -1 & 2 & \end{bmatrix}, \quad E_{ii} = \begin{bmatrix} 1 & -1 & & & & \\ -1 & 2 & -1 & & & \\ & \ddots & \ddots & \ddots & & \\ & & -1 & 2 & -1 & \\ & & & -1 & 1 & \end{bmatrix}.$$

Pre- and postmultiplication by C_i^T and C_i, respectively, means that we restrict ourselves to a space perpendicular to the constant. It holds, for E_{ii} and F_{ii} of order m_i and x perpendicular to a constant, that

$$\begin{aligned}\frac{4}{m_i^2}(x,x) \le (x, E_{ii}x) &\le 4(x,x),\\ 2(x,x) \le (x, F_{ii}x) &\le 6(x,x),\end{aligned}$$

where we have used that the smallest eigenvalue of E_{ii} on this space is $4\sin^2(\frac{\pi}{2m_i})$ and that $\sin(x) \ge 2x/\pi$ on $[0, \pi/2]$. So we have

$$\frac{2}{3m_i^2}(x, F_{ii}x) \le (x, E_{ii}x) \le 2(x, F_{ii}x).$$

Hence $\alpha_i = \frac{2}{3m_i^2}$ and $\beta_i = 2$. ∎

LEMMA 4.8. *Suppose* (9) *holds; then* $\rho(\hat{D}^{-1}\hat{R}) \le \sqrt{1-\alpha}$.

Proof. Note that $\hat{R}$ does not change if we replace A by $\breve{A}$, since $E_{11}e$ and $E_{22}e$ are zero. However, $\hat{D}$ does, so for the proof we add an argument to $\hat{D}$ to indicate this.

We have $\hat{D}(A) = \hat{D}(\breve{A}) + \text{diag}(C_1^T E_{11} C_1, C_2^T E_{22} C_2, 0, 0, 0, 0)$, and $([x;0], \hat{R}[0;y])^2 \leq ([x;0], \hat{D}(\breve{A})[x;0])([0;y], \hat{D}(\breve{A})[0;y])$. (We adopt the MATLAB notation ";" to indicate that $[x;y] = [x^T, y^T]^T$.) Using condition (9) we can now bound the first factor in the right-hand side

$$\begin{aligned}
([x;0], \hat{D}(\breve{A})[x;0]) = \quad & ([x;0], \hat{D}(A)[x;0]) \; - (x, \text{diag}(C_1^T E_{11} C_1, C_2^T E_{22} C_2)x) \\
\leq \quad & ([x;0], \hat{D}(A)[x;0]) \; - \alpha(x, \text{diag}(C_1^T F_{11} C_1, C_2^T F_{22} C_2)x) \\
\leq \quad & ([x;0], \hat{D}(A)[x;0]) \; - \alpha(x, \text{diag}(C_1^T F_{11} C_1, C_2^T F_{22} C_2)x) \\
& \qquad - \alpha([x;0], \hat{D}(\dot{A})[x;0]) \\
= \quad & \qquad (1-\alpha)([x;0], \hat{D}(A)[x;0]).
\end{aligned}$$

For the second factor it holds that $([0;y], \hat{D}(\breve{A})[0;y]) = ([0;y], \hat{D}(A)[0;y])$. This leads to a strengthened Cauchy–Bunyakovsky–Schwarz (CBS) inequality

$$([x;0], \hat{R}[0;y])^2 \leq (1-\alpha)([x;0], \hat{D}(A)[x;0])([0;y], \hat{D}(A)[0;y]).$$

This can now be used to get the desired bound

$$\begin{aligned}
([x;y], \hat{R}[x;y]) = \quad & ([x;0], \hat{R}[x;0]) + 2([x;0], \hat{R}[0;y]) + ([0;y], \hat{R}[0;y]) \\
= \quad & 2([x;0], \hat{R}[0;y]) \leq 2[\sqrt{1-\alpha}([x;0], \hat{D}(A)[x;0])([0;y], \hat{D}[0;y])] \\
\leq \quad & \sqrt{1-\alpha}[([x;0], \hat{D}(A)[x;0]) + ([0;y], \hat{D}[0;y])] \\
= \quad & \sqrt{1-\alpha}([x;y], \hat{D}[x;y]). \qquad \square
\end{aligned}$$

For the previous lemma we could use that R has Property A [36]. However, $\hat{D}_{aa}$ will not have this property for an arbitrary number of separators (it does in the above case of two separators). Fortunately, we can straightforwardly quantify the "diagonal dominance" of this matrix.

LEMMA 4.9. *Let* (9) *hold; then the eigenvalues of* $\breve{D}_{aa}^{-1}(\breve{D}_{aa} + \breve{R}_{aa})$ *are in the interval* $[\alpha, \frac{1}{\alpha}]$.

Proof.

$$\begin{aligned}
(x, (\breve{D}_{aa} + \breve{R}_{aa})x) &= (x, (\breve{D}_{aa}(\breve{A}) + \breve{R}_{aa})x) + (x, \text{diag}(C_1^T E_{11} C_1, C_2^T E_{22} C_2)x) \\
&\geq (x, \text{diag}(C_1^T E_{11} C_1, C_2^T E_{22} C_2)x) \\
&\geq \alpha(x, \text{diag}(C_1^T F_{11} C_1, C_2^T F_{22} C_2)x) + \alpha(x, (\breve{D}_{aa}(\dot{A}))x) \\
&= \alpha(x, \breve{D}_{aa}(A)x).
\end{aligned}$$

Hence a lower bound is α. For the upper bound we find

$$\begin{aligned}
(x, (\breve{D}_{aa} + \breve{R}_{aa})x) &= (x, (\breve{D}_{aa}(\dot{A}) + \breve{R}_{aa}(\dot{A}))x) + (x, \text{diag}(C_1^T F_{11} C_1, C_2^T F_{22} C_2)x) \\
&\leq (x, \text{diag}(C_1^T F_{11} C_1, C_2^T F_{22} C_2)x) \\
&\leq \frac{1}{\alpha}[(x, \text{diag}(C_1^T E_{11} C_1, C_2^T E_{22} C_2)x) + (x, (\breve{D}_{aa}(\breve{A}))x)] \\
&= \frac{1}{\alpha}(x, \breve{D}_{aa}(A)x). \qquad \square
\end{aligned}$$

This leads to the following main result.

THEOREM 4.10. *If condition* (9) *holds, then the preconditioned Schur complement of the Laplace matrix is bounded by* $(1+\sqrt{1-\alpha})^2/\alpha^3 < 4/\alpha^3$ *for an arbitrary number of separators and hence the convergence is independent of the grid size.*

Proof. Lemmas 4.6–4.9 can straightforwardly be generalized to an arbitrary number of separators. Furthermore $\alpha < 1$, $\beta = 1/\alpha$, $\eta = \sqrt{1-\alpha}$. Hence the condition number is according to Lemma 4.6 less than

$$\frac{(1+\eta)^2}{\alpha^2(1-\eta^2)} \leq \frac{4}{\alpha^3}.$$

α is a function of the m_i, so for arbitrarily large grids where all $m_i \leq m$, the same bound on the condition number holds. Hence the convergence behavior is independent of the grid size. ∎

If we take the α used in Lemma 4.7, the bound for the condition number is proportional to m^6. However, in practice we observe a much milder dependence. Therefore we studied how the condition number depends on the separator length for a simplified case of only one separator.

LEMMA 4.11. *For general SPD matrix A we have that the condition number can be bounded by $4/\sigma$ with $\sigma = (e,e)^2/[(e,Ae)(e,A^{-1}e)]$ and $\sigma \in (0,1]$.*

The proof of this lemma can be found in the appendix.

In the case of an approximation of the 2D Laplace equation with only one separator, the operator on the separator is spectrally equivalent to the square root of a one-dimensional Laplace operator with Dirichlet boundary conditions [5]. Therefore the eigenfunctions are sine functions. Using MAPLE we can now find that σ is proportional to $1/\log(1+m)$, where m is the number of unknowns on the separator. So for a special σ above we find that the condition number of the preconditioned matrix is proportional to $\log(1+m)$, which is the behavior we observe in the experiments.

We conclude this section by a number of remarks concerning the dropping strategy.

Scalar equations. The reader may have noticed that in this section we hardly mentioned the pressure. In fact, the combination of orthogonal transformations and dropping may also be applied to the pure diffusion problem. In section 5 we will start out by showing numerical results for the scalar Poisson equation.

The nonsymmetric case. One may ask how much of the above can be generalized to the nonsymmetric case (for instance, the Navier–Stokes equations). Assume the nonsymmetric matrix A is positive definite (PD), i.e., $(x, Ax) > 0$ for any nontrivial x. Then the Schur complement is PD and the orthogonal transformation does not destroy that property. Since all principle submatrices of a PD matrix are PD, the approximation will be PD. So the factorization will not break down. To say something about the condition number of the preconditioned matrix is more difficult. For a mild deviation from symmetry we expect the same behavior as for the symmetric case. However, the numerical results for the Navier–Stokes equations at relatively high Reynolds-numbers indicate that the method works very well even for highly nonsymmetric matrices.

Numerical stability. In traditional lumping, possible only for M-matrices, one simply lumps a coefficient on the diagonal. This means that a nonnegative matrix is subtracted. Eijkhout [7] showed already in the nineties that this may give a zero on the diagonal. This is easy to preclude by simply not allowing the diagonal to become zero. What is much harder to prevent is the occurrence of independent systems in the preconditioner, some of which may be singular. This easily occurs in anisotropic problems. The proposed dropping does not suffer from these problems.

Alternatives for H. Finally we propose a simple orthogonal extension to e in order to form H. Let m be the order of H and note that $[1,-1,0,\ldots,0]^T$, $[1,1,-2,0,\ldots,0]^T$,

$\ldots$, $[1, \ldots, 1, -(m-1)]^T, e$ are all orthogonal. They can be used for the extension after a proper scaling to the length of e. The application of this operator can be implemented by keeping a partial sum. In this way about $2m$ additions of rows of the matrix it is applied to are needed. The Householder transform has a similar operation count. One may ask whether alternative choices for C in H_1 influence the convergence. This is not the case. We can replace H_1 by $H_1 Q$. For arbitrary orthogonal matrices Q this has no influence on the analysis.

4.5. Iteration in the kernel of B^T. Since the fill of the B part remains at most 2 per row during the whole process, we will not drop there. This means that the B matrix is exact in the factorization, and with appropriate dropping (such as the strategy introduced in the previous section), the eigenvalues of the preconditioned matrix will all be positive and real. Still, we cannot directly apply the preconditioned conjugate gradient method since for that both original and preconditioner must be positive definite in the Krylov subspace. We can enforce this condition by building the Krylov subspace $\mathcal{K}(\tilde{K}^{-1}K, x)$ on a starting solution x that satisfies the constraint. In exact arithmetic $\mathcal{K}$ then remains in the kernel of B^T. In practice, accumulation of round-off errors will undermine this property.

This problem is often encountered in the field of constraint optimization, and Gould, Hribar, and Nocedal [12] have developed a variant of the conjugate gradient method, projected preconditioned CG (PPCG), which can be used for the Stokes problem. There are various ways to find a particular solution of $B^T v = b_2$, one of which is solving the system once, replacing K by the preconditioner.

For the Navier–Stokes equations one could devise a projected preconditioned FOM method, as long as the eigenvalues of the preconditioned matrix are in the right half-plane, but for the results shown in section 5.4 we simply used MATLAB's GMRES.

4.6. Computational complexity. We will now discuss the complexity of the algorithm, implemented as discussed in the previous sections. We assume that a direct method with optimal complexity is used for the solution of the relevant linear systems, so in three dimensions if the number of unknowns is $\mathcal{O}(N)$, the work is $\mathcal{O}(N^2)$, as with nested dissection. For the 3D (Navier–)Stokes equations, we have $N = \mathcal{O}(n^3)$ unknowns, where n is the number of grid cells in one space dimension. We keep the subdomain size constant and denote the number of unknowns per subdomain by $S = \mathcal{O}(s^3)$ (here s is the fixed separator length). Hence, there will be N/S subdomains. Per domain there will be $\mathcal{O}(s^2)$ non-V_Σ- and $\mathcal{O}(1)$ V_Σ-nodes. Per domain the amount of work required is as follows:

1. $\mathcal{O}(S^2)$ for the subdomain elimination;
2. transformation on faces with H: $\mathcal{O}(s^4)$;
3. factorization of non-V_Σ nodes: $\mathcal{O}((s^2)^3) = O(S^2)$.

The total over all domains is $\mathcal{O}(N/S)\mathcal{O}(S^2) = \mathcal{O}(NS)$, so in this part the number of operations decreases linearly with S (e.g., by a factor 8 if s is halved).

The solution of the reduced problem (V_Σ-nodes) requires $\mathcal{O}((N/S)^2)$ operations. Here doubling s will decrease the work by a factor 64. So in total the work per iteration is $\mathcal{O}(NS) + \mathcal{O}((N/S)^2)$. The number of iterations is constant for S constant. There is, however, a positive dependence on S as we may expect. In the next section we will observe that the number of iterations is proportional to $\log(1 + S)$. So if we double s, aproximately a fixed amount of iterations is added.

It is clear that if we solved the reduced problem iteratively by applying our method recursively until the problem has a fixed grid-independent size, the overall complexity would be $\log(1 + S)\mathcal{O}(NS)$.

5. Numerical experiments. In this section we will demonstrate the performance of the new solver by applying it to a series of increasingly complex problems relevant to computational fluid dynamics. For each problem we first keep the subdomain size constant while refining the mesh. As discussed in the previous section, the complexity of the algorithm will then be linear in the number of unknowns except when solving the reduced Schur complement: the operations required to factor a single subdomain matrix stays the same and the number of subdomains increases linearly with the grid size. Furthermore, both size and connectivity pattern of the separators remain the same so the amount of work per separator remains constant while the number of separators increases linearly, too.

The second experiment will be to fix the grid size and vary the subdomain size (i.e., the number of subdomains). The expectation here is that due to fill-in the bulk of the workload shifts from the Schur complement toward the subdomain factorization as the size of the subdomains is increased.

For each experiment, the following data is displayed:

- n_x: the grid size is $n_x \times n_x$ ($n_x \times n_x \times n_x$) in 2D (3D), respectively.
- s_x: the subdomain size is $s_x \times s_x$ ($s_x \times s_x \times s_x$) in 2D (3D), respectively.
- N: number of unknowns (size of the saddle point matrix),
- nnz: number of nonzeros in original matrix,
- $\mathrm{N_S}$: number of unknowns on the separators and remaining p's (size of the Schur complement),
- n: number of $V_\Sigma's$ and remaining p's (size of reduced Schur complement),
- iter: number of CG iterations performed on the Schur complement to reduce the residual norm by $1/\mathrm{tol} = 10^8$,
- fill 1: grid-independent part of relative fill-in (number of nonzeros in the solver divided by number of nonzeros in original matrix). The grid-independent portion consists of
  (a) fill-in generated while factoring the subdomain matrices,
  (b) fill-in generated while constructing the Schur complement,
  (c) fill-in generated while factoring the separator-blocks of the preconditioner,
- fill 2: grid-dependent part of relative fill-in, generated when factoring the $n \times n$-dimensional reduced Schur complement.
- κ: condition estimate of the preconditioned Schur complement: fraction of the largest and smallest eigenvalue (by magnitude) of the generalized eigenvalue problem $\mathrm{S}x + \lambda \mathrm{M}x = 0$, where S is the Schur complement, and M the preconditioner used. We use approximations to the actual eigenvalues computed by MATLAB's "eigs" command (not all tables contain this value).

Remark. The fill listed under fill 1 (b) can be avoided by not explicitly constructing the Schur complement. The fill listed as fill 2 grows with increasing grid size, but it can be made grid-independent by solving S_2 iteratively, too (i.e., by applying our method recursively).

We do not show plots of the convergence behavior. Since all the results are obtained by CG the convergence is, apart from the first few digits gained, completely regular, which shows that the eigenvalues, except for a few outliers at the beginning, appear in a cluster. The relatively stringent convergence tolerance of eight digits ensures that the overall convergence behavior does not strongly depend on the choice of the initial vector. Choosing a smaller tolerance results in stagnation for some of the tests below because the conditioning of the matrix does not allow for more accurate

solutions.

The general behavior we observe in the second experiment is that the number of iterations scales with $\log(1 + s_x)$, where s_x is the separator length. So doubling the separator length means an increase of the number of iterations by a constant amount.

5.1. The Poisson equation. We first investigate Poisson's equation, discretized using second order central differences on a regular structured grid (standard five point and seven point stencils in two dimensions and three dimensions, respectively). This is an important case as solving Poisson's equation is central to most CFD problems, for instance, to determine the pressure in explicit time stepping algorithms. Tables 1 and 2 show the 2D results. The first shows the dependence on grid refinement and the latter the influence of the domain sizes. Similar results for the 3D case are shown in Tables 3 and 4.

TABLE 1
2D Poisson-equation: grid refinement, subdomain size $s_x = 8$.

n_x	N	nnz	N_S	n	iter	fill 1	fill 2	κ
32	1 024	5 112	240	48	21	5.53	0.20	7.04
64	4 096	20 472	960	192	21	5.52	0.39	7.04
128	16 384	81 912	3 840	768	21	5.52	0.68	7.04
256	65 536	327 672	15 360	3 072	21	5.52	1.03	7.04
512	262 144	1 310 712	61 440	12 288	21	5.52	1.59	7.04
1 024	1 048 576	5 242 872	245 760	49 152	21	5.52	2.20	7.04

TABLE 2
2D Poisson-equation: increasing subdomain size, grid size $n_x = 1024$.

s_x	N	nnz	N_S	n	iter	fill 1	fill 2	κ
4	1 048 576	5 242 872	458 752	196 608	16	2.01	11.5	4.00
8	1 048 576	5 242 872	245 760	49 152	21	5.52	2.29	7.04
16	1 048 576	5 242 872	126 976	12 288	27	9.84	0.39	11.2
32	1 048 576	5 242 872	64 512	3 072	32	13.8	0.063	16.5

TABLE 3
3D Poisson-equation: grid refinement, subdomain size $s_x = 8$.

n_x	N	nnz	N_S	n	iter	fill 1	fill 2	κ
16	4 096	28 660	1 352	56	24	29.7	0.064	10.1
32	32 768	229 364	10 816	448	25	29.0	0.36	10.2
64	262 144	1 834 996	86 528	3 584	25	29.0	1.53	-

TABLE 4
3D Poisson-equation: increasing subdomain size, grid size $n_x = 64$. ($\kappa^\star$ computed at $n_x = 32$.)

s_x	N	nnz	N_S	n	iter	fill 1	fill 2	$\kappa^\star$
4	262 144	1 834 996	151 552	28 672	19	3.68	52.0	5.75
8	262 144	1 834 996	86 528	3 584	25	29.0	1.5	10.2
16	262 144	1 834 996	46 144	448	30	116.2	0.045	16.7

5.2. Darcy's law. For flows in porous media one often has to solve the Darcy problem, where A is just a diagonal matrix. One approach is to eliminate the velocities, which leads to a Poisson equation. Care has to be taken when calculating the velocities, because the gradient operator has to be applied to the pressure. In this numerical differentiation of the pressure field, round-off errors may be amplified too much to obtain an accurate solution. Therefore, Darcy's problem is often solved in primitive form. Tables 5–8 show the numerical results for Darcy's law in two and three space dimensions.

TABLE 5
2D Darcy-equation: grid refinement, subdomain size $s_x = 8$.

n_x	N	nnz	N_S	n	iter	fill 1	fill 2	κ
16	736	2 400	65	17	16	5.53	0.061	3.77
32	3 008	9 920	385	109	25	6.29	0.24	10.8
64	12 160	40 320	1 793	533	26	6.65	0.49	12.2
128	48 896	162 560	7 681	2 341	26	6.82	1.00	12.6
256	196 096	652 800	31 745	9 797	26	6.91	1.69	12.6
512	785 408	2 616 320	129 025	40 069	26	6.95	2.64	12.7
1 024	3 143 680	10 475 520	520 193	162 053	26	6.97	3.58	-

TABLE 6
2D Darcy-equation: increasing subdomain size, grid size $n_x = 512$.

s_x	N	nnz	N_S	n	iter	fill 1	fill 2	κ
8	3 143 680	10 475 520	520 193	162 053	26	6.97	3.58	12.7
16	3 143 680	10 475 520	258 049	40 069	29	11.1	0.66	17.6

TABLE 7
3D Darcy-equation: grid refinement, subdomain size $s_x = 4$.

n_x	N	nnz	N_S	n	iter	fill 1	fill 2	κ
8	1 856	6 720	492	171	34	10.8	1.28	14.0
16	15 616	57 600	5 878	2 683	36	10.2	17.6	15.3
32	128 000	476 160	54 762	27 819	36	9.73	87.7	15.4
40	251 200	936 000	109 972	56 971	36	9.65	167.	-

TABLE 8
3D Darcy-equation: increasing subdomain size, grid size $n_x = 40$. ($\kappa^\star$ computed at $n_x = 32$.)

s_x	N	nnz	N_S	n	iter	fill 1	fill 2	$\kappa^\star$
4	251 200	936 000	109 972	56 971	36	9.65	167.	15.4
8	251 200	936 000	53 037	11 601	39	50.2	16.7	18.3

5.3. A Stokes problem. The problem is a 2D Stokes equation on the unit square

$$\left.\begin{aligned} -\nu\Delta\mathbf{u} + \nabla p &= 0\,, \\ \nabla\cdot\mathbf{u} &= 0 \end{aligned}\right\}, \tag{10}$$

where $\mathbf{u}(x, y)$ is the velocity field and $p(x, y)$ the pressure field; the parameter ν controls the amount of viscosity. We can get rid of the parameter ν by defining a new pressure variable $\bar{p} = p/\nu$. If the first equation is divided by ν, we can substitute p by $\bar{p}$ and the parameter ν is gone. So we may assume that $\nu = 1$.

These equations are discretized on a uniform staggered grid (a C-grid, see Figure 1) which results in an $\mathcal{F}$-matrix. It is singular because the pressure field is determined up to a constant.

For the Stokes problem the matrix B^T represents the discrete divergence operator. Consequently, we call the kernel of this matrix the divergence free space. As a solution of this problem we choose a random vector in the divergence free space. So the right-hand side of the divergence equation is zero in our case.

We start off the iteration with the zero vector (which is trivially in the divergence free space) and therefore we can use the projected conjugate gradient method (see section 4.5). Results are summarized in Tables 9 through 12.

TABLE 9
2D Stokes-equation: grid refinement, subdomain size $s_x = 8$.

n_x	N	nnz	N_S	n	iter	fill 1	fill 2	κ
16	736	4 196	65	17	18	7.79	0.057	4.93
32	3 008	17 604	385	109	27	8.39	0.25	12.8
64	12 160	72 068	1 793	533	31	8.68	0.65	13.8
128	48 896	291 588	7 681	2 341	31	8.72	1.33	14.2
256	196 096	1 172 996	31 745	9 797	31	8.70	2.40	14.6
512	785 408	4 705 284	129 025	40 069	31	8.60	3.83	15.0

TABLE 10
2D Stokes-equation: increasing subdomain size, grid size $n_x = 512$.

s_x	N	nnz	N_S	n	iter	fill 1	fill 2	κ
4	785 408	4 705 284	260 097	162 053	24	3.65	20.0	9.6
8	785 408	4 705 284	129 025	40 069	31	8.60	3.83	15.0
16	785 408	4 705 284	63 489	9 797	38	15.7	0.60	21.9

TABLE 11
3D Stokes-equation: grid refinement, subdomain size $s_x = 4$.

n_x	N	nnz	N_S	n	iter	fill 1	fill 2	κ
8	1 856	13 728	492	171	34	13.9	1.20	16.6
16	15 616	122 304	5 878	2 683	41	12.5	16.4	23.8
32	128 000	1 029 504	54 762	27 819	43	11.5	103.	27.1
40	251 200	2 030 880	109 972	56 971	43	11.3	168.	-

TABLE 12
3D Stokes-equation: increasing subdomain size, grid size $n_x = 40$. ($\kappa^\star$ computed at $n_x = 32$.)

s_x	N	nnz	N_S	n	iter	fill 1	fill 2	$\kappa^\star$
4	251 200	2 030 880	109 972	56 971	43	11.3	167.	27.1
8	251 200	2 030 880	53 037	11 601	49	65.8	12.1	39.1

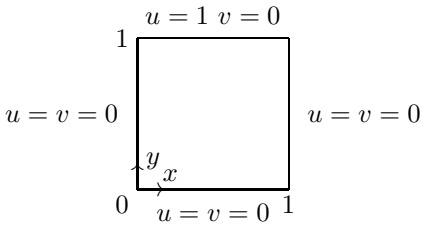

FIG. 3. *Geometry for the lid-driven cavity problem.*

5.4. Incompressible flow in a lid-driven cavity. As test problem for the Navier–Stokes equations we use the lid-driven cavity. In [21] this problem was studied near the transition point from steady to transient flow. The stability of steady and periodic solutions was investigated using the Newton–Picard method [17] with the θ-method for time stepping (with θ slightly larger than 0.5 in order to damp high-frequency modes which would otherwise show up as spurious eigenvalues near the imaginary axis). The linear systems that have to be solved have a slightly increased diagonal, which improves the conditioning somewhat. The matrix renumbering ILU (MRILU) preconditioner [4] used at the time converged slowly and not at a grid-independent rate. In a recent review [9], the performance of a number of block multilevel preconditioners is investigated for the steady problem for Reynolds numbers up to 1000. These methods also solve the coupled equations, but perform inner iterations on the velocity and pressure part separately and hence require many parameters to be tuned. Below we demonstrate robust, grid-independent convergence for the driven cavity problem at Reynolds numbers of up to 8000.

The problem consists of calculating the flow in a square cavity with uniformly moving lid. The domain and boundary conditions of the lid-driven cavity problem are shown in Figure 3, where u and v denote the velocity in x- and y-direction, respectively.

The equations are given by

$$\left.\begin{aligned} -\mathbf{u}\cdot\nabla\mathbf{u} + \tfrac{1}{Re}\Delta\mathbf{u} - \nabla p &= 0\ , \\ \nabla\cdot\mathbf{u} &= 0 \end{aligned}\right\}. \tag{11}$$

For the discretization we use a symmetry-preserving space discretization [26], which is stable and does not introduce artificial diffusion. Furthermore, the grid is stretched towards the boundaries in order to resolve the boundary layers. The ratio between largest and smallest mesh size is about 5. This also means that we really need to change to fluxes through grid cell boundaries instead of velocities in order to get the required property that all elements in B have the same magnitude (see the beginning of section 4). The convergence tolerance is set to 10^{-6} in these experiments. The system matrix is the Jacobian from the first step of the Newton method at the current Reynolds number. In order to avoid convergence problems of Newton's method, we use the result at the previous Reynolds number as a starting solution (the Reynolds numbers used are shown in Table 13).

We first focus on the effect of increasing the Reynolds number (cf. Table 13). The convergence is not independent of the Reynolds number. In our view this is not surprising, because the underlying continuous problem changes with the Reynolds number and more and more eigenvalues are getting close to the origin. This is different from the dependence on the mesh, where the continuous problem stays the same and all eigenvalues near the origin stay at their place.

Next we refine the grid at a high Reynolds number of 8000, close to the point (cf. [21]) where the steady state becomes unstable; results are shown in Table 14. Note that the number of iterations is going down as we decrease the mesh size. This is because with decreasing mesh size the physical size of the subdomains is decreasing if we keep the number of unknowns per subdomain the same. As the physical subdomain decreases, the diffusion plays a more important role than the advection on that scale. Since the approximations take place at the subdomain scale, the convergence behavior tends to that of the Stokes problem.

TABLE 13
2D driven cavity: increasing Reynolds number, grid size $n_x = 512$.

Re	N	nnz	N_S	n	iter	fill 1	fill 2
500	785 408	6 794 252	129 025	40 069	59	6.41	2.59
1000	785 408	6 794 252	129 025	40 069	73	6.39	2.59
2000	785 408	6 794 252	129 025	40 069	87	6.38	2.65
4000	785 408	6 794 252	129 025	40 069	104	6.35	2.78
8000	785 408	6 794 252	129 025	40 069	130	6.33	2.72

TABLE 14
2D driven cavity: grid refinement at $Re = 8000$.

n_x	N	nnz	N_S	n	iter	fill 1	fill 2
64	12 160	103 820	1 793	533	185	6.09	0.418
128	48 896	420 620	7 681	2 341	181	6.22	0.953
256	196 096	1 693 196	31 745	9 797	167	6.29	1.75
512	785 408	6 794 252	129 025	40 069	130	6.33	2.72

We mention that with the resulting preconditioner it was also quite easy to compute eigenvalues using MATLAB's eigs routine (i.e., ARPACK). Hence we can now study the stability problem near the point where the steady state becomes unstable using eigenvalue analysis. In order to give the reader an idea of the computational cost of the method, we conclude by presenting some preliminary timing results using a C++ implementation of the algorithm in Table 15. In two dimensions the setup phase is dominated by the construction of the Schur complement, which may be implemented in a more efficient way. In parallel it may be more efficient not to construct the Schur complement explicitly but to use its definition when applying it in the Krylov sequence. In three dimensions the dominating steps in the two-level method are the factorization of the reduced Schur complement (V_Σ-nodes) and its solution in every iteration. This would be alleviated in a multilevel method. A detailed study of the performance of the method using a parallel implementation will be presented in future work.

6. Discussion and conclusions. In this paper we have shown that the structure preserving complete LDL^T factorization introduced in [19] of an $\mathcal{F}$-matrix can be transformed into an incomplete factorization. We constructed an iterative solver for the whole system, which avoids having to balance inner and outer iterations as in a segregated approach. Depending only on a single parameter (the subdomain size), the method is as easy to use as a direct solver and gives reliable results in a reasonable turn-around time.

For Stokes matrices we were able to prove grid-independent convergence. The

TABLE 15

Timing results for the 2D Stokes problem on a regular C-grid, subdomain size $s_x = 8$. Thirty-eight GMRES iterations are performed to reach an accuracy of 10^{-8}. All times are given in seconds.

Grid size	32	64	128	256	512
complete setup	2.0e-01	1.0e+00	5.0e+00	2.4e+01	2.0e+02
complete solve	5.5e-02	1.6e-01	6.7e-01	2.9e+00	1.2e+01
factor interior variables	4.6e-02	1.9e-01	9.2e-01	3.7e+00	1.5e+01
form Schur complement	2.5e-02	1.4e-01	8.0e-01	5.9e+00	1.2e+02
transform and drop	5.1e-02	2.9e-01	1.4e+00	6.2e+00	2.6e+01
factor non-V_Σs	3.9e-03	1.9e-02	9.3e-02	2.5e-01	1.0e+00
factor V_Σs	3.0e-03	8.3e-03	3.4e-02	2.7e-01	1.3e+00
solve interior	1.0e-02	5.5e-02	2.4e-01	9.7e-01	3.9e+00
solve non-V_Σs	5.7e-03	1.7e-02	8.0e-02	4.1e-01	1.7e+00
solve V_Σs	1.2e-02	2.1e-02	8.8e-02	4.2e-01	1.9e+00

total number of operations required is currently not grid-independent since we use a direct solver to solve the reduced system. However, the amount of work required for this step is reduced by about the cube of the subdomain size in two dimensions and the sixth power in three dimensions. So increasing the subdomain size by a factor 2 means in two dimensions a factor 8 and in three dimensions a factor 64. For the Navier–Stokes equations we also observed grid-independent convergence. We are developing a parallel C++ implementation of the method that can be applied recursively, making it a multilevel method.

We proved the robustness of the method for Stokes and Navier–Stokes equations, where in the latter case the matrix should be definite. Computations show that the method still performs well for cases where eigenvalues pass the imaginary axis away from the origin (Hopf bifurcations).

In the case of $\mathcal{F}$-matrices we are able to keep the computation in the kernel of the constraint equation, i.e., for Stokes in the divergence free space, allowing us to use the CG method. Though the $\mathcal{F}$-matrices seem to be a limited class due to the constraints on the sparsity pattern in B, many applications lead to matrices of this type.

The next generalization would be a discretization of Stokes on an Arakawa B-grid as used in oceanography [27] . Here the velocities are located in the same points in space, while the pressure is in the middle of four velocities. The pressure gradient in a velocity point is now computed as the mean of a difference, using all four pressures around the velocity. This results in a B matrix with four entries per row instead of two. However, due to the special construction of the differences, it is possible to cast it in the above framework. First results are similar to the ones shown above.

In principle the method can also immediately be applied to Schur complement systems that are obtained from domain decomposition of (Navier–)Stokes problem where per domain only one average pressure is left such as, e.g., described in [22, p. 262].

The present method is just an approximation of the associated direct method. In order to find generalizations it is easiest to first generalize the direct method. After that, the iterative method can be derived along the same lines as the one presented here.

Appendix. Proof of Lemma 4.11. After transformation we find the matrix

$$\begin{bmatrix} C^TAC & C^TAe \\ eTAC & e^TAe \end{bmatrix}.$$

The symmetrized block Jacobi iteration matrix associated to this matrix is

$$\begin{bmatrix} 0 & E \\ E^T & 0 \end{bmatrix},$$

where $E = (C^TAC)^{\frac{1}{2}}C^TAe(e^TAe)^{\frac{1}{2}}$. If the spectral radius of this matrix is less than γ, then we have that the condition number of the preconditioned matrix is bounded by

$$(1+\gamma)/(1-\gamma). \tag{12}$$

The spectral radius is less than one if we can find a $\gamma < 1$ such that

$$\left|\left(\begin{bmatrix} x \\ y \end{bmatrix}, \begin{bmatrix} 0 & E \\ E^T & 0 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix}\right)\right| \leq \gamma\left(\begin{bmatrix} x \\ y \end{bmatrix}, \begin{bmatrix} x \\ y \end{bmatrix}\right),$$

which reduces to $2|(x, Ey)| \leq \gamma((x,x)+(y,y))$. Squaring both sides we can derive that this is the case if $(x, Ey)^2 \leq \gamma^2(x,x)(y,y)$. If we expand E, we can see that this is equivalent to finding γ from the maximization

$$\gamma^2 = \max_x \frac{(Cx, Ae)^2}{(Cx, ACx)(e, Ae)}.$$

Since Cx will just give a vector perpendicular to e we can also solve

$$\gamma^2 = \max_{x \perp e} \frac{(x, Ae)^2}{(x, Ax)(e, Ae)}. \tag{13}$$

Without loss of generality we can take $(x, Ax) = 1$. Thus the optimization problem can be turned into an unconstrained one using Lagrange multipliers

$$\gamma^2 = \max_{x,\eta,\mu,} \frac{(x, Ae)^2}{(e, Ae)} - \eta\{(x, Ax) - 1\} + 2\mu(e, x).$$

The stationary point has to be found from the equations

$$\frac{(x, Ae)}{(e, Ae)}(e_i, Ae) - \eta(e_i, Ax) + \mu(e_i, e) = 0,$$

together with the constraints $(x, Ax) = 1$ and $(e, x) = 0$. We add an extra unknown c which ought to be equal to $(x, Ae)/(e, Ae)$. This leads to the following eigenvalue problem:

$$\begin{bmatrix} -\eta A & e & Ae \\ e^T & 0 & 0 \\ e^TA & 0 & -(e, Ae) \end{bmatrix}\begin{bmatrix} x \\ \mu \\ c \end{bmatrix} = 0.$$

We omit the condition (x, Ax), since, after solving the eigenvalue problem, we can scale the eigenvector such that this condition is satisfied. Solving this we find

$$\eta = 1 - \frac{(e, e)^2}{(e, Ae)(e, A^{-1}e)}.$$

The associated singular vector is defined by $\mu = -\rho c$, where $\rho = (e,e)/(e,A^{-1}e)$ and $\eta Ax = c(-\rho e + Ae)$. Substituting this x into (13) one obtains

$$\begin{aligned}\gamma^2 &= \frac{(\rho(e,e) - (e,Ae))^2}{(e,Ae)[\rho^2(e,A^{-1}e) - 2\rho(e,e) + (e,Ae)]} \\ &= \frac{(\rho(e,e) - (e,Ae))^2}{(e,Ae)[-\rho(e,e) + (e,Ae)]} \\ &= \frac{(e,Ae) - \rho(e,e)}{(e,Ae)} = 1 - \rho(e,e)/(e,Ae) = 1 - \sigma,\end{aligned}$$

where $\sigma = (e,e)^2/[(e,Ae)(e,A^{-1}e)]$. Since $(e,e) = (e,A^{-1}Ae)$ we have that $\sigma \leq 1$. So $\gamma = \sqrt{1-\sigma}$.

Using (12) we can now bound the condition number

$$\frac{1+\sqrt{1-\sigma}}{1-\sqrt{1-\sigma}} = \frac{(1+\sqrt{1-\sigma})^2}{\sigma} \leq \frac{4}{\sigma}.$$

Acknowledgments. Part of this work was done during a sabbatical leave of the first author to the Numerical Analysis Group at Rutherford Appleton Laboratory in Chilton (UK). We kindly thank that group for the hospitality and good fellowship.

REFERENCES

[1] M. Arioli and G. Manzini, *Null space algorithm and spanning trees in solving Darcy's equation*, BIT, 43 (2003), pp. 839–848.
[2] M. Benzi, G.H. Golub, and J. Liesen, *Numerical solution of saddle point problems*, Acta Numer., 14 (2005), pp. 1–137.
[3] M. Benzi and M.A. Olshanskii, *An augmented Lagrangian-based approach to the Oseen problem*, SIAM J. Sci. Comput., 28 (2006), pp. 2095–2113.
[4] E.F.F. Botta and F.W. Wubs, *Matrix renumbering ILU: An effective algebraic multilevel ILU preconditioner for sparse matrices*, SIAM J. Matrix Anal. Appl., 20 (1999), pp. 1007–1026.
[5] P.E. Bjørstad and O.B. Widlund, *Iterative methods for the solution of elliptic problems on regions partitioned into substructures*, SIAM J. Numer. Anal., 23 (1986), pp. 1097–1120.
[6] J.H. Bramble and J.E. Pasciak, *A domain decomposition technique for Stokes problems*, Appl. Numer. Math., 6 (1990), pp. 251–261.
[7] V. Eijkhout, *Beware of unperturbed modified incomplete factorizations*, in Iterative Methods in Linear Algebra, R. Beauwens and P. de Groen, eds., Elsevier Science, North–Holland, Amsterdam, 1992, pp. 583–591.
[8] H.C. Elman, D.J. Silvester, and A.J. Wathen, *Performance and analysis of saddle point preconditioners for the discrete steady-state Navier-Stokes equations*, Numer. Math., 90 (2002), pp. 665–688.
[9] H. Elman, V.E. Howle, J. Shadid, R. Shuttleworth, and R. Tuminaro, *A taxonomy and comparison of parallel block multi-level preconditioners for the incompressible Navier-Stokes equations*, J. Comput. Phys., 227 (2008), pp. 1790–1808.
[10] J. Gaidamour and P. Hénon, *A parallel direct/iterative solver based on a Schur complement approach*, in Proceedings of the IEEE 11th International Conference on Computational Science and Engineering, Sao Paulo, Brazil, 2008, pp. 98–105.
[11] A. George and J.W.H. Liu, *Computer Solution of Large Sparse Positive Definite Systems*, Prentice-Hall Series in Computational Mathematics, Prentice-Hall, Englewood Cliffs, NJ, 1981.
[12] N.I.M. Gould, M.E. Hribar, and J. Nocedal, *On the solution of equality constrained quadratic programming problems arising in optimization*, SIAM J. Sci. Comput., 23 (2001), pp. 1376–1395.
[13] P. Hénon and Y. Saad, *A parallel multistage ILU factorization based on a hierarchical graph decomposition*, SIAM J. Sci. Comput., 28 (2006), pp. 2266–2293.
[14] G. Karypis and V. Kumar, *A fast and high quality multilevel scheme for partitioning irregular graphs*, SIAM J. Sci. Comput., 20 (1998), pp. 359–392.

[15] D. Kay, D. Loghin, and A.J. Wathen, *A preconditioner for the steady-state Navier–Stokes equations*, SIAM J. Sci. Comput., 24 (2002), pp. 237–256.
[16] C. Keller, N.I.M. Gould, and A.J. Wathen, *Constraint preconditioning for indefinite linear systems*, SIAM J. Matrix Anal. Appl., 21 (2000), pp. 1300–1317.
[17] K. Lust, D. Roose, A. Spence, and A.R. Champneys, *An adaptive Newton–Picard algorithm with subspace iteration for computing periodic solutions*, SIAM J. Sci. Comput., 19 (1999), pp. 1188–1209.
[18] A.C. de Niet and F.W. Wubs, *Two saddle point preconditioners for fluid flows*, Int. J. Numer. Methods Fluids, 54 (2007), pp. 355–377.
[19] A.C. de Niet and F.W. Wubs, *Numerically stable LDL^T-factorization of F-type saddle point matrices*, IMA J. Numer. Anal., 29 (2009), pp. 208–234.
[20] Y. Notay, *An aggregation-based algebraic multigrid method*, Electron. Trans. Numer. Anal., 37 (2010), pp. 123–146.
[21] G. Tiesinga, F.W. Wubs, and A.E.P. Veldman, *Bifurcation analysis of incompressible flow in a driven cavity by the Newton-Picard method*, J. Comput. Appl. Math., 140 (2002), pp. 751–772.
[22] A. Toselli and O. Widlund, *Domain Decomposition Methods—Algorithms and Theory*, Springer-Verlag, Berlin, 2005.
[23] M. Tůma, *A note on the LDL^T decomposition of matrices from saddle-point problems*, SIAM J. Matrix Anal. Appl., 23 (2002), pp. 903–915.
[24] S.A. Vavasis, *Stable numerical algorithms for equilibrium systems*, SIAM J. Matrix Anal. Appl., 15 (1994), pp. 1108–1131.
[25] H.A. van der Vorst, *Iterative Krylov Methods for Large Linear Systems*, Cambridge Monogr. Appl. Comput. Math. 13, Cambridge University Press, Cambridge, UK, 2003.
[26] R.C.W.P. Verstappen and A.E.P. Veldman, *Symmetry-preserving discretization of turbulent flow*, J. Comput. Phys., 187 (2003), pp. 343–368.
[27] F.W. Wubs, A.C. de Niet, and H.A. Dijkstra, *The performance of implicit ocean models on B- and C-grids*, J. Comput. Phys., 211 (2006), pp. 210–228.
[28] D.M. Young, *Iterative Solution of Large Linear Systems*, Academic Press, New York, London, 1971.